COMPEND 2DX3 – Microprocessor Systems Project
Final Project Report

Sapna Suthar – suthas4 – 400513077

Instructor: Dr. Yaser M. Haddara, Dr. Shahrukh Athar, Dr. Thomas Doyle

Department of Electrical and Computer Engineering, McMaster University

Due: April 9, 2025

Table of Contents

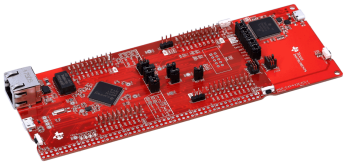
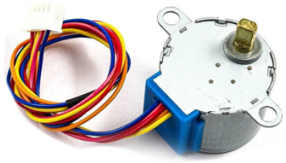
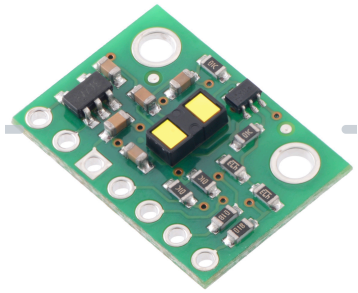
Design Overview

This device was created as an inexpensive, small-scale version of a LiDAR (Light Detection and Ranging) system. It was made to be portable for use in mapping out indoor spaces in 3D. The base of the system is the Texas Instrument MSP-EXP432E401Y microcontroller. This works with an onboard button, 3 onboard LEDs, the ULN2003 driver board, the 28BYJ-48 stepper motor and the Pololu VL53L1X Time of Flight (ToF) sensor.

Overview

- A 3D spatial map is generated and visualized using MATLAB.
- The Time-of-Flight (ToF) sensor transmits real-time distance data to the microcontroller via the I2C protocol.
- The microcontroller sends this data live to a PC over UART at a baud rate of 115200 bps.
- Powered through a 5V micro-USB connection.
- Measurement progress is indicated using an LED on PF0, with an additional status LED on PN0.
- The microcontroller operates at a 10 MHz bus speed.
- Two onboard push buttons are used—one to rotate the stepper motor and the other to initiate ToF data acquisition and transmission.
- ADC sampling rate follows the Nyquist criterion: $f_{sample} \geq 2f_{signal}$

Hardware

Component	Image	Datasheet	Price
MSP-EXP432E401Y Microcontroller		MSP-EXP432E401Y Datasheet	$47.99
28BYJ-48 Stepper Motor		Stepper Motor Datasheet	$24.85
Pololu VL53L1X		Pololu VL53L1X Datasheet	$11.39

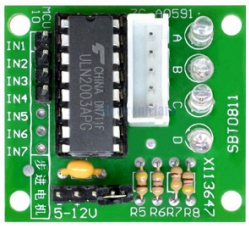
ULN2003 Stepper Motor Driver		Stepper Motor Driver Datasheet	$1.95

Table 1: Hardware Components

Software

Software	Use	Link
MATLAB	3D Mapping Visualization	MathWorks Sign In
Keil MDK (C)	Microcontroller Functions	Keil

Table 2: Software Components

Features

Below are the key performance features of the individual hardware components used in this design:

MSP-EXP432E401Y Microcontroller

- ARM Cortex-M4 32-bit CPU with floating-point support
- Maximum CPU clock: 120 MHz
- Design bus speed: 10 MHz
- Interfaces: JTAG, Micro USB, Ethernet
- 1 MB flash memory, 256 kB SRAM
- I2C support with high-speed mode
- UART at 115200 bps for real-time data communication
- Utilizes 1 onboard user buttons with debouncing and pull-up/down resistors
- Utilizes 3 user-controllable LEDs
- GPIO pins support interrupts, PWM, and ADC

28BYJ-48 Stepper Motor

- Unipolar 4-phase stepper motor
- Standard stride angle: 5.625° (64 steps per revolution)
- Utilizes 11.25° (32 steps per revolution)
- Configured to run in full-step mode
- Operating voltage range: 5–12 VDC

ULN2003 Stepper Motor Driver

- Accepts 5–12 V power input
- Includes onboard 4-channel signal outputs
- Uses Texas Instruments ULN2003AN driver IC

Pololu VL53L1X Time-of-Flight (ToF) Sensor

- I2C communication at 100 kbps
- Up to 27° field of view
- Operating voltage: 2.6V – 5.5V
- Emits 940 nm invisible Class 1 VCSEL laser
- Three ranging modes:
    - Short: Up to ~130 cm
    - Medium: Up to ~300 cm
    - Long: Up to ~400 cm

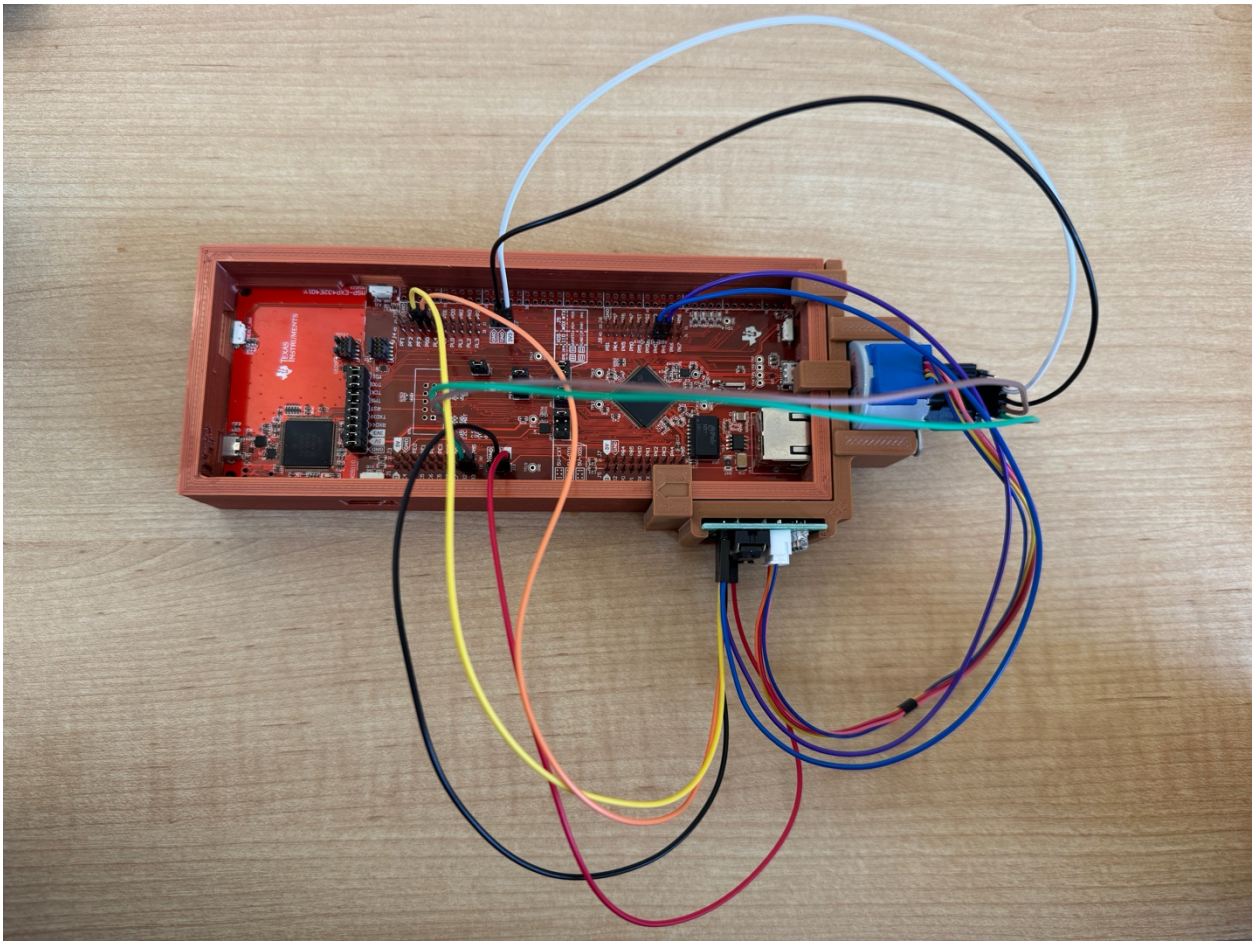


Figure 1: Physical Implementation of the LiDAR System

General Description

The goal of this device was to create an inexpensive, portable version of a LiDAR (Light Detection and Ranging) systems for indoor use.

The 3D LiDAR system was created using the MSP-EXP432E401Y microcontroller, 28BYJ-48 stepper motor, Pololu VL53L1X Time of Flight (ToF) sensor and the ULN2003 stepper motor driver. The MSP-EXP432E401Y microcontroller was the centre of the system and interfaced with the stepper motor and its driver and the ToF sensor. The microcontroller has 120MHz, 32-bit ARM Cortex M4F CPU, 1MB of Flash, 25kB of SRAM and floating-point capabilities. For this specific system, a bus speed of 10MHz was utilized. The system begins with a user pressing the onboard button PJ0. When it is pressed the stepper motor begins to rotate and scan at increment.

The 28BYJ-48 is a 4-phase unipolar stepper motor. This stepper motor receives its power from the ULN2003 stepper motor driver. The stepper motor is activated when the onboard button PJ0 is pressed. The motor runs on 5V and has a stride angle of 5.625° per 64 steps. For this system being built, the stepper motor rotates 360° (1 full rotation) in increments of 11.25°, stopping 32 times to acquire measurements.

The Pololu VL53L1X Time of Flight (ToF) sensor is a laser-based distance sensor. It utilizes a 940 nm Class 1 laser along with a SPAD (single photon avalanche diode) array to detect reflected light. It operates on a supply voltage between 2.6V and 5V and can measure distances of up to 4 meters. In this system, the sensor is activated by pressing PJ0 and takes a distance reading every 11.25° of rotation. Although the ToF sensor initially detects analog signals, the onboard circuitry handles the entire analog-to-digital conversion process. This includes transduction, signal conditioning, and conversion into a digital format that can be processed by a microcontroller. Communication between the sensor and the microcontroller is handled via the I2C protocol, where SDA serves as the data line and SCL as the clock line. In this project, the sensor transmits the distance measured back to the microcontroller.

After each distance measurement is obtained, the distance is transmitted using UART to a computer to being the visualization process. In this system, the only parameter transmitted was distance. The visualization program in MATLAB receives the distance through one of the computer's communication ports and parses that data. The MATLAB program then converts the distance and angle to cartesian coordinates (y, z). The x coordinate is preset as the scan count starting at 0. Using the (x, y, z) coordinates, MATLAB plots the data on a 3D scatter plot connecting the appropriate points.

Block Diagram

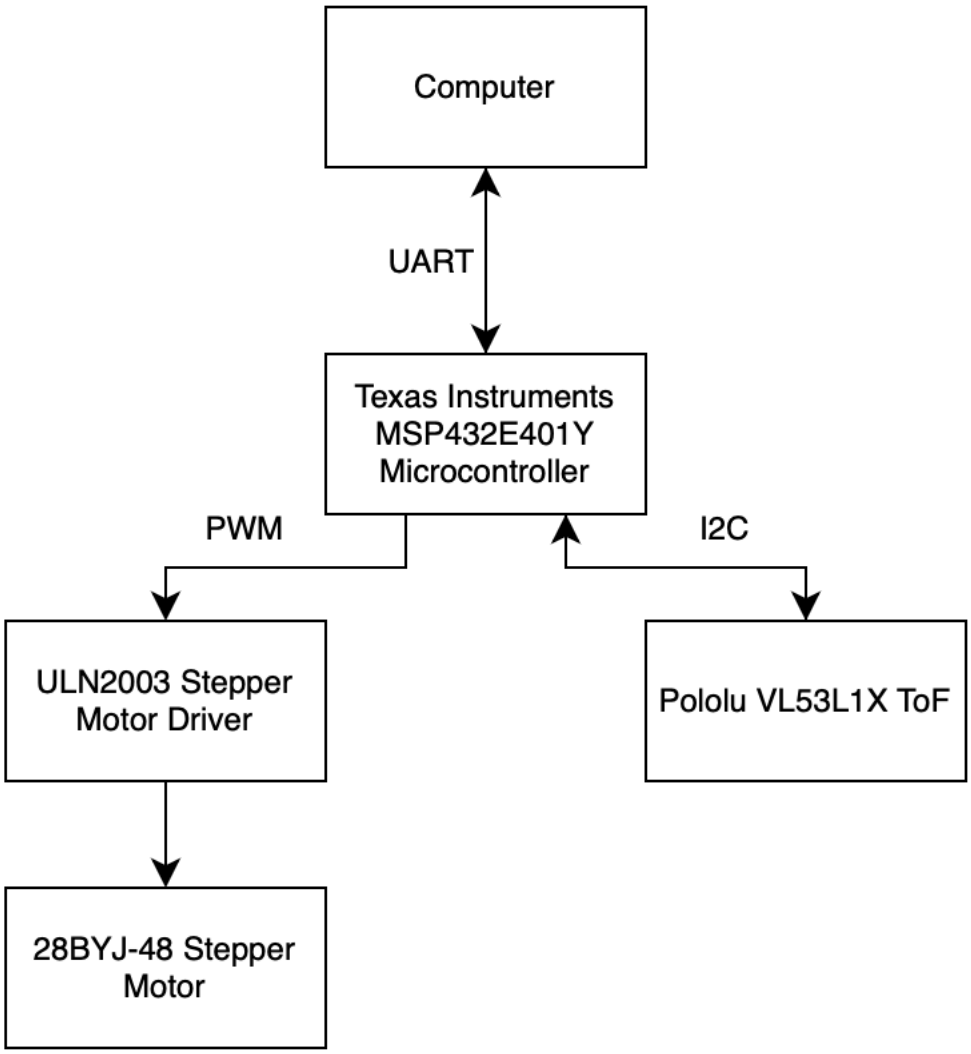


Figure 2: Component Block Diagram

Data Flow Graph

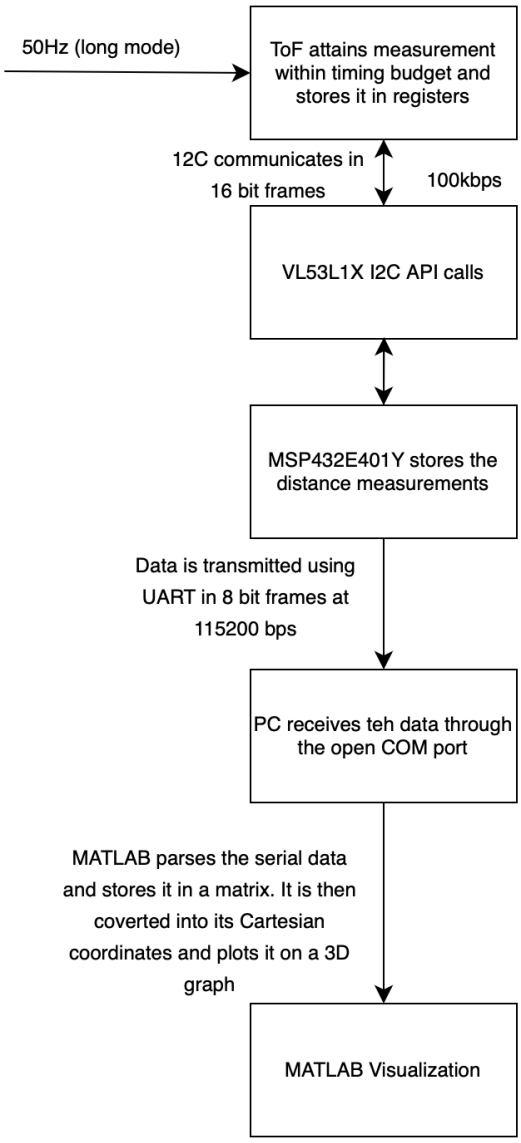


Figure 3: Data Flow Graph

Device Characteristics Table

Component	Power Requirements (V)
MSP-EXP432E401Y Microcontroller	+5V from MicroUSB cable
ULN2003 Stepper Motor Driver	+5V from MSP-EXP432E401Y Microcontroller
28BYJ-48 Stepper Motor	+5V from ULN2003 Stepper Motor Driver
Pololu VL53L1X ToF	+3.3V from MSP-EXP432E401Y Microcontroller

Table 3: Component Power Requirements

MSP-EXP432E401Y Pin	Signal Description	Use
GPIO Port H [H0:H3]	PWM	Output for Stepper Motor Driver
GPIO Port J [J0]	GPIO Polling	
GPIO Port N [N0:N1]	Toggle/Flash	Onboard LED D1, D2
GPIO Port F [F4]	Toggle/Flash	Onboard LED D3
GPIO Port E [E0]	PWM	Bus speed verification using SysTick()
GPIO Port B2	SCL	I2C Clock Line
GPIO Port B3	SDA	I2C Data Line
MicroUSB	UART	Outputs data to computer through UART protocol

Table 4: MSP-EXP432E401Y Pinout Descriptions

Parameter	Value
I2C Frame	1 start bit 7 follower address bits 1 read/write bit 1 acknowledge bit 8 data bits 1 acknowledge bit 1 stop bit
I2C Clock	100 kbps
I2C Address VL53L1X	0x29
UART Frame	1 start bit 8 data bits 1 stop bit (no parity)
UART Baud Rate	115200 bps

Table 5: Communication Protocol Parameters

Parameter	Value
MSP-EX432E401Y Bus Speed	10MHz
Bus Speed Check Period	2ms

Table 6: Timing Parameters

Detailed Description

Distance Measurement

The VL53L1X Time of Flight distance sensor was used to obtain distance measurements. The sensor measures distance using a 940 nm Class 1 infrared laser and a SPAD (Single Photon Avalanche Diode) receiver array. The sensor works by emitting an infrared laser pulse that reflects off nearby surfaces and returns to the receiver. The SPAD functions similarly to a traditional diode and operates in reverse bias, just below its breakdown voltage. When a photon enters this region, it generates an electron-hole pair. The electric field created by the reverse bias accelerates these charge carriers, and if the photon's energy is sufficient, it triggers an avalanche effect. This means it produces more electron-hole pairs and creates a detectable current. This mechanism allows the sensor to detect the returning light with high sensitivity and calculate the time it took to return. This is then used to determine the distance to the target.

Based on the above process, the ToF sensor calculates distance in millimetres based on the known speed of light and time delay between the emission and return of photons. To determine the distance, the equation used is $\Delta d = c \cdot \frac{\Delta t}{2}$, where c is the speed of light (approximately 300,000 km/s) and Δt is the total time between the laser being emitted and the reflected photons being detected by the SPAD. The time includes the journey to the object and back, so it is divided by 2 to determine the distance one way. The calculated distance is transmitted to the microcontroller using I2C communication protocol.

After powering the microcontroller, the Keil Code is loaded onto the microcontroller and stored in the flash memory to immediately run. The code begins with initializing the GPIO pins, LEDs, communications protocols, the system clock and the ToF. Once initialized, the ToF is set to continuous ranging mode and long mode. Then the microcontroller enters a loop where it monitors the state of the onboard push button PJ0. The onboard button toggles a global flag "isScanning". Once, scanning is enabled, the stepper motor begins to rotation and collects a distance measurement from the ToF sensor every 16 steps (11.25°). During clockwise scanning, the measured distance is transmitted through UART to the computer for processing. The onboard LEDs indicate system activity. In this system, PN0 flashes at the beginning of a scanning revolution, PN1 blinks with each measurement and PF4 is on whiles the scan is active. After a full 360° scan (512 steps) is complete, the motor spins in the opposite direction for a full revolution without collecting data. Then it is ready for the next scan cycle to begin.

To capture a full 360° scan, the VL53L1X ToF sensor was mounted on to the 28BYJ-48 stepper motor, which rotates in the yz-plane. The stepper motor is driven by energizing specific coils in a defined sequencing using full steps. Based on the 28BYJ-48 stepper motor, one full rotation (360°) requires 512 steps. This means that each step is $\frac{360^\circ}{512\ steps} = 0.703125^\circ$. For this project a distance measurement was taken every 11.25° causing a measurement to be taken every $\frac{0.703125^\circ}{1\ step} = \frac{11.25^\circ}{x\ steps} \Rightarrow \frac{11.25^\circ}{0.703125^\circ} = 16\ steps$. This results in 32 samples per full revolution, enabling a high-resolution 3D scan. The motor movement and data collection are controlled within the motor() function. During each clockwise scan, a counter tracks the motor steps, and every time the counter reaches a multiple of 16, the system pauses to collect a distance

measurement from the ToF sensor and sends it to the computer via UART. LED indicators help the user track system behavior: PN0 flashes at the start of each scan, PN1 blinks every time a measurement is taken, and PF4 remains on during scanning. After completing 512 steps (360°), the motor stops, waits, then rotates counterclockwise back to its original position without taking measurements. This counterclockwise return helps prevent wire tangling and prepares the system for the next scan cycle.

Once the global isScanning flag is set to true by pressing the onboard button PJ0, the scanning process begins within the motor() function. The stepper motor advances one step at a time, and every 16 steps (equivalent to 11.25°), the system checks if the ToF sensor has new data available. The sensor is polled using the VL53L1X API to ensure data readiness before retrieving a valid distance measurement. This value is then transmitted to a PC via UART in millimeters. The corresponding scan angle is calculated by multiplying the step count by 0.703125° per step. The system uses this to construct a 2D profile of the scanned environment in the YZ-plane. The distance value is the only parameter sent through UART, while the step and angle tracking are managed internally within the MATLAB code. Once a full 360° scan (512 steps) is complete, the motor reverses direction and returns to its starting point without collecting data. This ensures clean rewinding and prepares the system for the next scan cycle. Communication between the microcontroller and the PC is handled over UART through a micro-USB connection.

Visualization

The visualization process is handled by a MATLAB program. The program beings by configuring the serial communication with the microcontroller. It does so by setting the COM port, baud rate and clearing any serial buffer data. For each scan, 32 distance measurements spread evenly across 1 rotation are taken. These measurements are spaced at 11.25° intervals. The program initializes an array to store the Cartesian coordinates. The x coordinate is the scan number starting from 0. The y coordinate is the cosine projected distance. The z coordinate is the sine projected distance. Each scan layer has a fixed x value, while y and z are computed using the following equations:

$$y = distance \times \cos(\theta)$$
$$z = distance \times \sin(\theta)$$

As the program waits for the microcontroller to transmit 32 distance values, each received value is parsed and appended to the scan array. Once a full scan is received, it is converted to Cartesian coordinates, which are then added to the full dataset. After all the scans are completed, the coordinates are reshaped into matrices of dimensions $steps\ per\ scan \times number\ of\ scans$ to prepare for 3D visualization. Then MATLAB's 3D plotting functions are then used to construct the final graph. The visualization consists of connecting the adjacent points in one scan ring (constant x value) and connecting the corresponding points with the same angle and varying depth (x value). This creates a mesh-like 3D representation of the scanned environment.

Application Example

This section will explain the steps through an exemplar scan of a 3D space using the LiDAR system. Below are the setup instructions for initializing the system and acquiring measurements. As well, an expected output from the system can be seen for comparison with my assigned location. Based on my student number, the assigned area was location H. This location is found inside McMaster's Engineering Technology Building (ETB) on the second floor. A map of ETB's second floor is provided below along with reference photos of the scanned location. The instructions below assume that any needed software is already downloaded and configured, and the necessary steps have been taken to connect the hardware together. The required software can be seen in the Device Overview section and the hardware connections can be seen in the Circuit Schematic section.

Instructions

1. Connect the MSP-EXP432E401Y microcontroller to your computer using a Micro USB cable. Determine the COM port the two devices are communicating through. The port can be found through the Windows Device Manager and will be labeled XDS110 Class Application/User UART (COM#)
2. Open the MATLAB script and alter the following parameters based on your use
    a. Change `port = "COM4";` to the communication port being used by your computer and the microcontroller (found in Step 1).
    b. Change `num_scans = 3;` to correspond to the number of depth measurements (full 360° revolutions) you want to take.
    c. Change `values_per_scan = 32;` to match the desired number of measurements your ToF will take for a single 360° rotation. In this application example, 32 measurements were taken, meaning a measurement was taken every 11.25°. The more measurements taken, the more accurate the scan will be.
3. Ensuring the microcontroller is wired correctly (check Circuit Schematic section), open the Keil project in the Keil IDE.
4. After ensuring the desired target settings are correctly setup, in the top left corner of Keil click: $Translate \rightarrow Build \rightarrow Load$
5. After the project is flashed onto the microcontroller, press the reset button next to the microUSB.
6. Go to the MATLAB script and press the RUN button. Nothing should happen because we haven't started sending data yet.
7. Return to Keil and press the onboard button PJ0 to run the code and start sending data. This should turn LED D4 on. The stepper motor should begin spinning and the ToF will begin transmitting data. To check, ensure distances are being printed out in MATLAB's Command Window
8. Once the stepper motor has completed one full rotation, it will pause, do a full rotation in the opposite direction and pause again. During this time, you can move the LiDAR system one unit forward.
9. After the desired number of scans has been measured, MATLAB will automatically create a 3D scatter plot of the scanned area.

Scanned Location

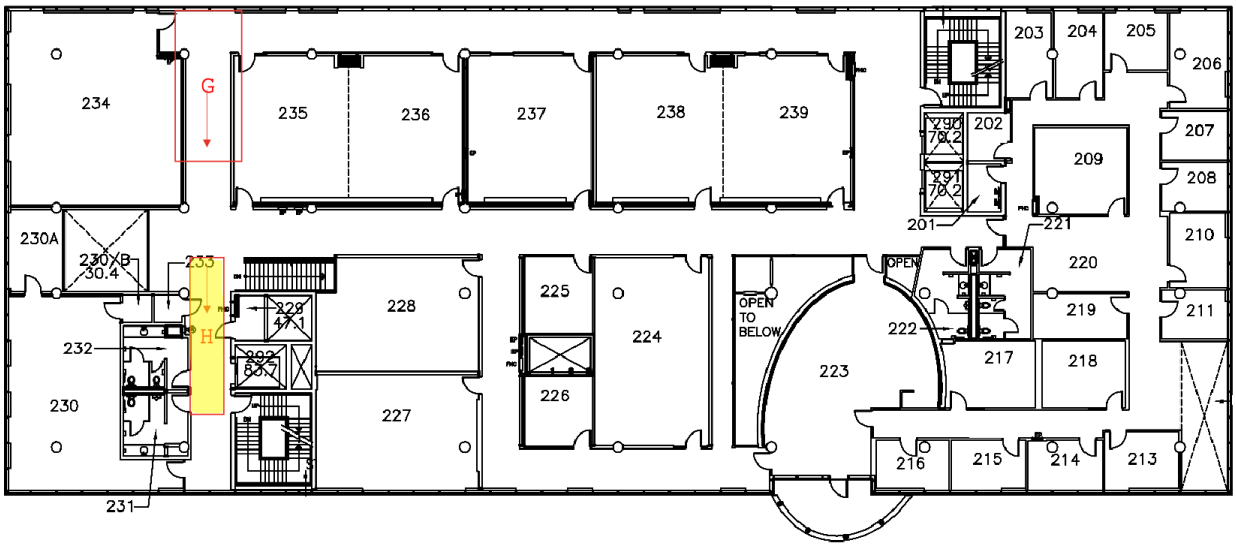


Figure 4: Engineering Technology Building (ETB) Second Floor

Reference Images

Figure 5: Reference photos of the scanned hallway

Expected Output (MATLAB Visualization)

From the scans below, the system was able to capture the general shape and some finer details. For example, in *Figure 6* the front view showcases the clear change in the height of the ceiling. Aswell, in *Figure 7*, the slight protrusions on the left side are very visible, showing the indent where the elevator doors are.

Note the scan was taken starting at an angle, so the hallway is slanted in the 3D simulation.

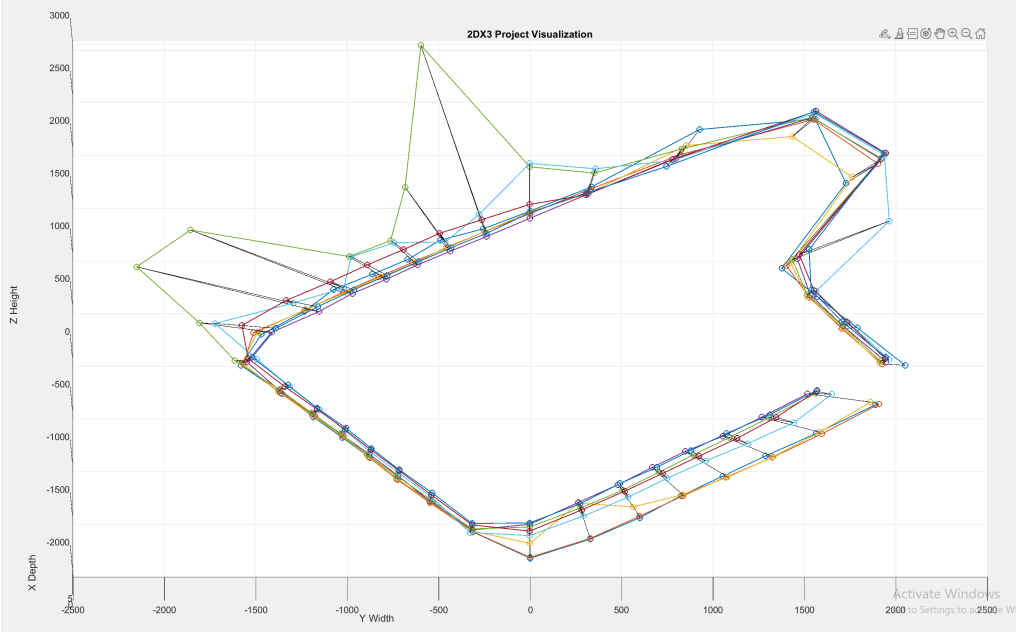


Figure 6: Front View of the Hallway Scan

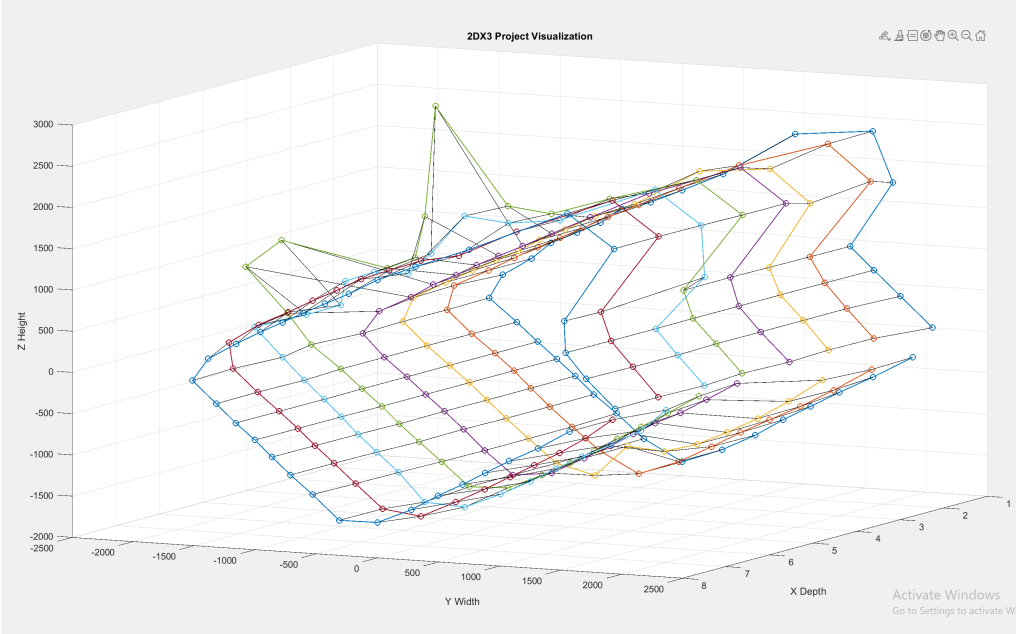


Figure 7: Right View of the Hallway Scan

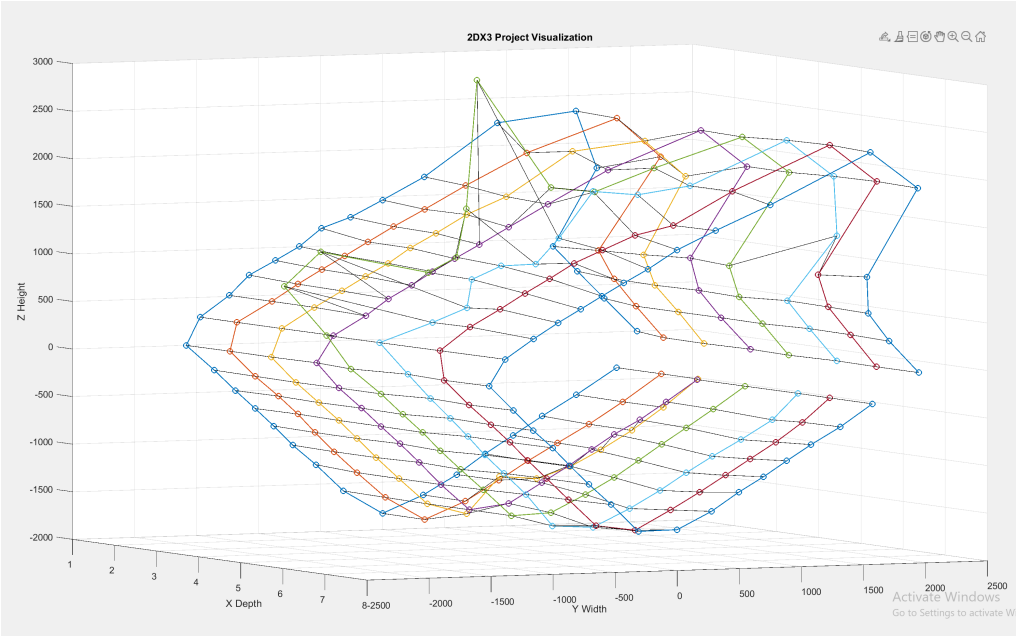


Figure 8: Left View of the Hallway Scan

Remember that the measurements from the ToF are in the yz plane and the x-axis is manually created through incrementing the number of scans.

Limitations

This system is intended to be a low-cost, adaptable alternative to traditional LIDAR systems, utilizing a stepper motor and VL53L1X Time-of-Flight (ToF) sensor for spatial data collection. However, given the constraints of the components and the real-time serial communication requirements, there are several hardware limitations that impact system performance and data accuracy:

- Due to cable routing and wire tension, the stepper motor must reverse its direction after each full 360° scan to prevent wire tangling. This adds delays as the system must wait for the motor to return to its original position before beginning a new scan.
- The stepper motor has a non-accumulative step error of ±5%.
- In extended use, the motor was prone to overheating
- The motor used can only reliably step in increments of 0.703125°. This means the accuracy of the scan is limited by the minimum step angle.
- The motor requires at least a 2ms delay between step activations to function reliably.
- The VL53L1X has a maximum reliable measurement range of 4 meters under ideal conditions. The range is further reduced when detecting non-reflective or highly textured surfaces, which may yield inconsistent or inaccurate results.
- The VL53L1X uses a timing budget (20–1000ms) that directly impacts accuracy and range. A longer timing budget improves measurement reliability but increases total scan duration. To ensure consistent results, the stepper motor must pause between steps long enough for the ToF sensor to complete its measurement cycle.
- The sensor data is encoded in 16-bit resolution. Using the formula below, we estimate the maximum quantization error: $Max\ Quanitzation\ Error = \frac{4m}{2^{16}} = 0.061035$
- The VL53L1X communicates over I2C, which is limited to a maximum of 400 kHz (fast mode). This design uses a 100kbps I2C transfer rate.
- One of the primary bottlenecks in system speed is the combined delay introduced by stepper motor rotation and the timing budget required by the VL53L1X ToF sensor.
- UART serial communication between the microcontroller and PC was configured at 128000bps (the highest rate verified via Windows 11's Device Manager). This was confirmed by checking the 'Port Settings' of the 'XDS110 Class Application/User UART' device on COM 4, where 128000bps was the maximum selectable option.
- The MSP432E401Y features a 32-bit floating point unit (FPU) for single-precision operations (add, subtract, multiply, divide, trigonometric). However, it's slower and less accurate than the 64-bit FPUs in modern PCs. To avoid delays, trigonometric operations (e.g., polar to Cartesian conversion) are handled in the MATLAB script on the PC instead of on the microcontroller.
- Although MATLAB efficiently handles Cartesian conversion and 3D plotting, these processes depend on receiving complete and correctly ordered serial data from the microcontroller. Any delay or corruption in the serial stream may require re-scanning or script interruption.

Circuit Schematic

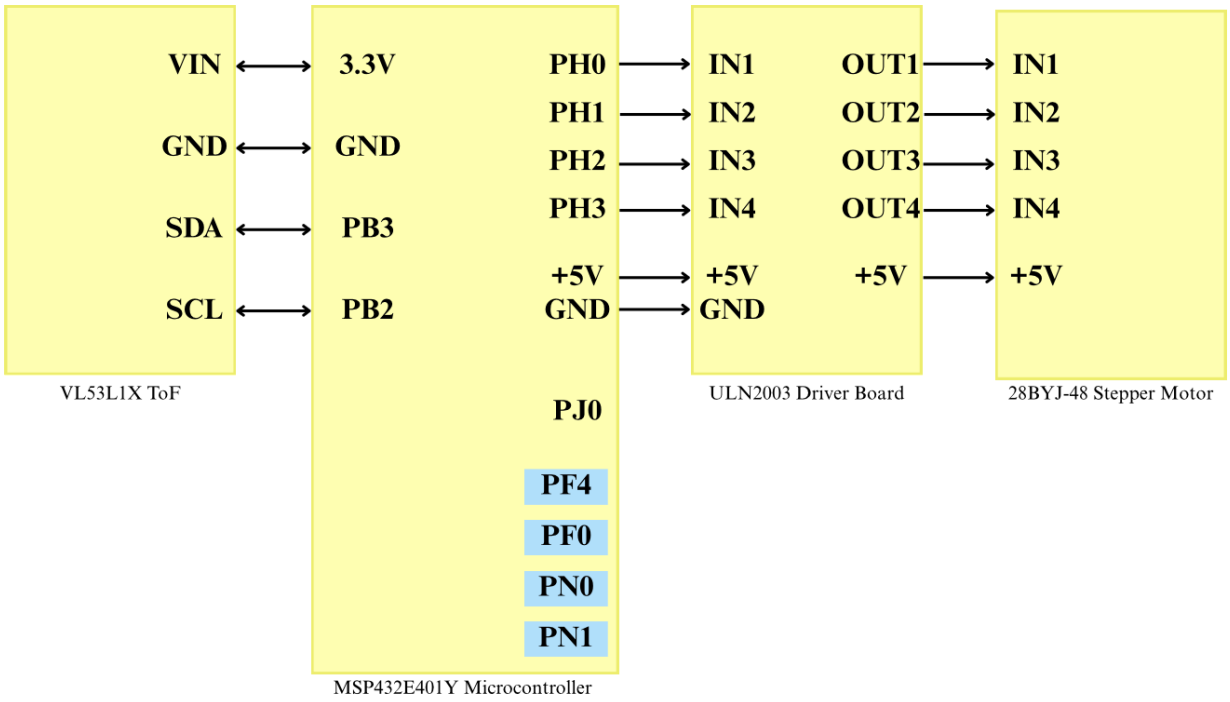


Figure 9: LiDAR System Circuit Diagram

Programming Logic Flowcharts

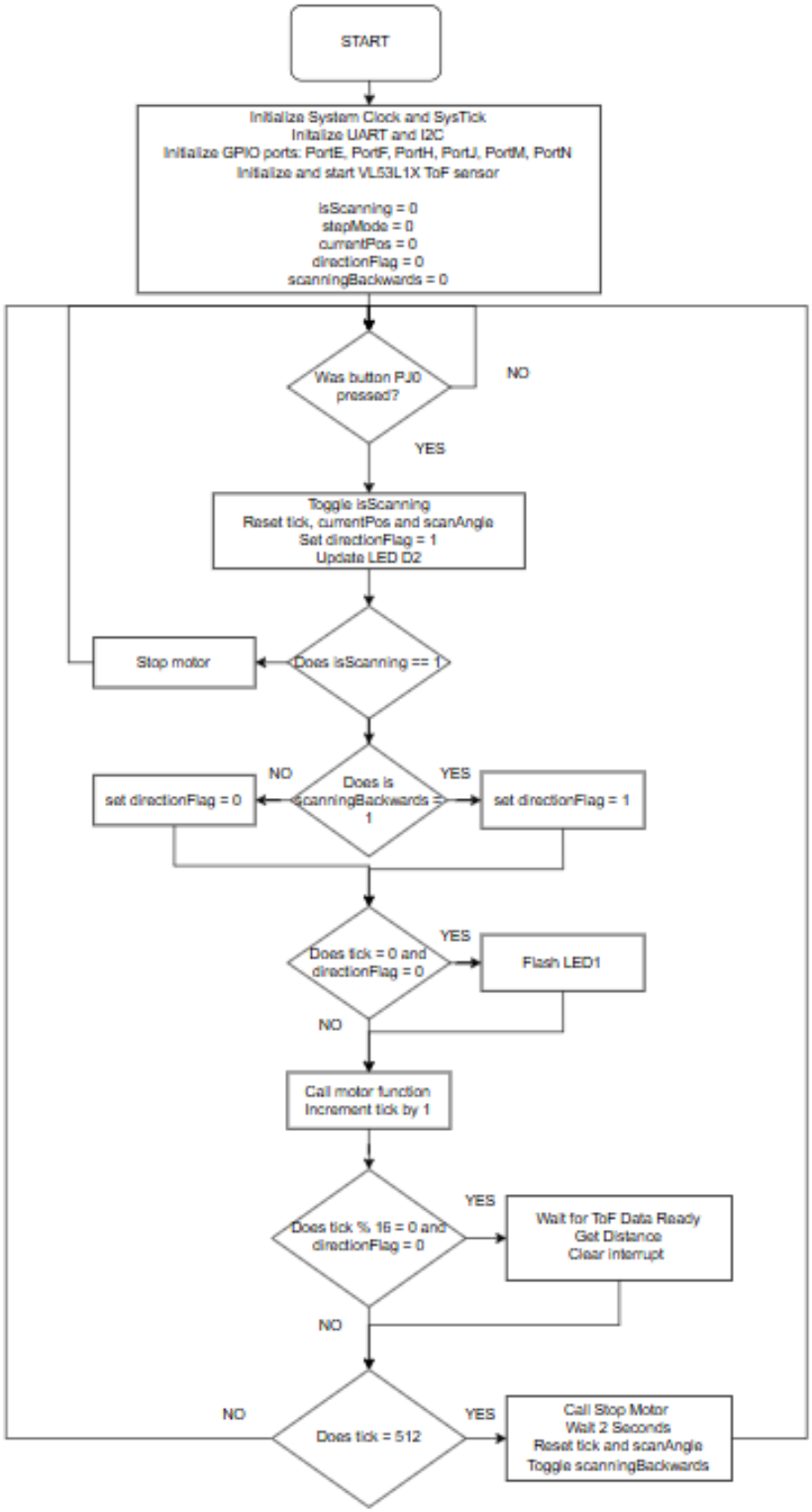


Figure 10: Keil Code Flowchart

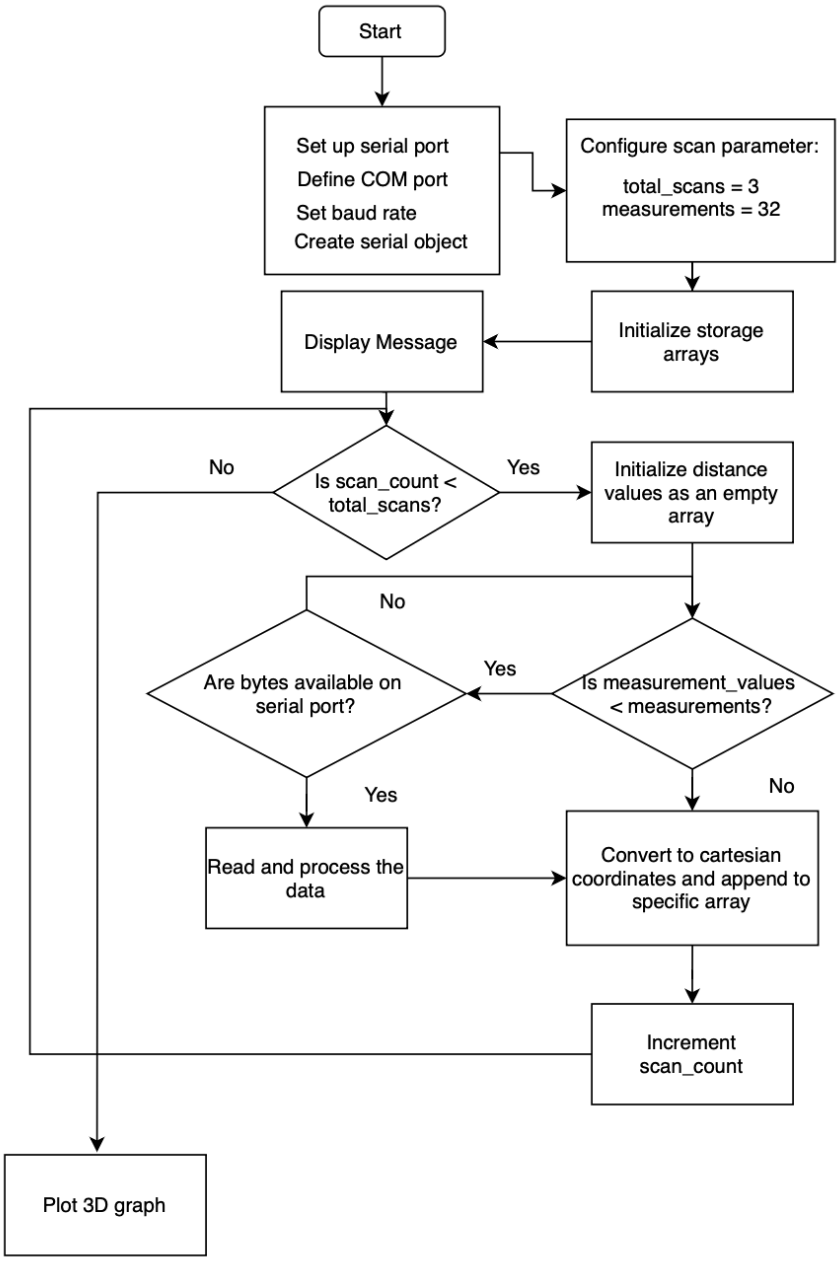


Figure 11: MATLAB Visualization Flowchart

References

[1] “VL53L1X,” Pololu, https://www.pololu.com/file/0J1506/vl53l1x.pdf [accessed Apr. 3, 2025].

[2] “MSP432E4 SimpleLinkTM Microcontrollers Technical Reference Manual,” Texas Instruments, https://www.ti.com/lit/ug/slau723a/slau723a.pdf [accessed Apr. 3, 2025].